A 3D Data Transformation Processor

Dimitrios Megas, Kleber Pizolato, Timothy Levin, and Ted Huffmire
Department of Computer Science
United States Naval Postgraduate School
Monterey, California 93943
{dmegas,kpizolat,levin,tdhuffmi}@nps.edu

ABSTRACT

Application-specific coprocessors, including those for cryptography and compression, can provide significant acceleration and power savings to programs requiring their services. While most coprocessors have traditionally been constructed as a separate chip connected to the main CPU over a relatively slow bus connection, 3D integration, providing a more direct connection, is an emerging technology that offers significant performance advantages and power savings over such systems. With 3D integration, two or more dies can be fabricated separately and later combined into a 3D integrated circuit (3D IC), a single stack of two or more dies connected by vertical conductive posts.

We propose a novel coprocessor architecture in which one layer houses application-specific coprocessors for cryptography and compression, which provide acceleration for applications running on a general-purpose processor in another layer. A compelling application for such a system is one that performs real-time trace collection, compressing the trace prior to its transmission to permanent off-chip storage for offline program analysis. Furthermore, an optional encryption step, performed by the cryptographic circuitry in the coprocessor layer, can protect this compressed data from interception. In another application, a high-performance stand-alone encryption service can be provided.

Categories and Subject Descriptors

B.7.1 [**Hardware**]: Integrated Circuits: Types and Design Styles—*Advanced Technologies*

General Terms

Design, Economics, Measurement, Performance, Security

Keywords

3D Integration, Cryptographic Hardware, Compression, Profiling

1. INTRODUCTION

We present a 3D architecture for the real-time transformation (compression or encryption) of a stream of data. A 3D IC data transformation processor is useful for collecting execution traces, e.g., for reverse engineering of malicious software, and post-mortem analysis of a system that has suffered an attack. Because of the reduced wire length made possible by stacking, a 3D architecture offers latency advantages over traditional coprocessors that are packaged separately and connected at the circuit board level or traditional 2D chips that combine a CPU and a coprocessor on the same die. The CPU layer, or *computation plane*, can be sold to ordinary customers without the coprocessor layer, or *control plane*, attached, but customers with high trustworthiness or high performance requirements can purchase the joined unit. Moreover, the coprocessor layer, alone, can be manufactured in a trusted foundry to provide the requisite trustworthiness to the combined system, foregoing the expense of using a trusted foundry for the CPU layer. This approach has the potential to improve the economic feasibility of trustworthy system acquisition.

For each design parameter for our proposed design, we justify our choices based on analysis of real 3D systems and 2D data transformation processors described in the literature. We also used binary instrumentation to generate trace files from the computation plane, which we then compress in order to compare the compression ratios for a variety of design variables and trace compression algorithms. Key decision factors for our design include:

- Semiconductor manufacturing process (e.g., 45nm)
- The components in the control plane
- The type of interface between the two dies
- The method of coordination between the two dies
- Type of communication interface within the control plane
- Method of delivery of I/O and power
- Size, type, and number of computation plane components
- Method of clock synchronization between planes

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE
OCT 2012

2. REPORT TYPE

3. DATES COVERED
00-00-2012 to 00-00-2012

4. TITLE AND SUBTITLE
A 3D Data Transformation Processor

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School,Department of Computer Science,Monterey,CA,93943

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
Proceedings of the Workshop on Embedded Systems Security (WESS)," Tampere, Finland, October 2012

14. ABSTRACT
Application-specific coprocessors, including those for cryptography and compression, can provide significant acceleration and power savings to programs requiring their services. While most coprocessors have traditionally been constructed as a separate chip connected to the main CPU over a relatively slow bus connection, 3D integration, providing a more direct connection, is an emerging technology that offers significant performance advantages and power savings over such systems. With 3D integration, two or more dies can be fabricated separately and later combined into a 3D integrated circuit (3D IC), a single stack of two or more dies connected by vertical conductive posts. We propose a novel coprocessor architecture in which one layer houses application-specific coprocessors for cryptography and compression, which provide acceleration for applications running on a general-purpose processor in another layer. A compelling application for such a system is one that performs real-time trace collection, compressing the trace prior to its transmission to permanent off-chip storage for offline program analysis. Furthermore, an optional encryption step, performed by the cryptographic circuitry in the coprocessor layer, can protect this compressed data from interception. In another application, a high-performance stand-alone encryption service can be provided.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**10**	

2. BACKGROUND

3D integration is an emerging chip fabrication technique in which multiple integrated circuit dies are joined using conductive posts. 3D integration offers several performance and security advantages, including extremely low latency and high bandwidth between the two dies and the ability to augment a processor with tightly-coupled custom features. Other advantages include lower power consumption, the ability to join disparate technologies, and the ability to control the lineage of a subset of the dies, e.g., by using a trusted foundry.

Dies can be bonded using face-to-face or face-to-back techniques. With face-to-face bonding, the metal layers of the two dies are facing each other, and die-to-die vias are used to connect the two layers. Only two dies can be joined in this fashion. With face-to-back bonding, the bulk silicon substrate (back) of one die is joined with the metal layers (face) of the other die, and through-silicon vias (TSVs) connect the metal layers of both dies. TSVs are relatively larger than face-to-face vias. Face-to-back bonding can join more than two dies.

Traditional 2D cryptographic coprocessors can be connected to general-purpose processors at the circuit-board level, or, in a multi-core system-on-chip (SoC), at the chip level. Some processors include cryptographic functions in the instruction set architecture (ISA); however, the ISA is frozen, whereas with 3D, a wide range of custom cryptographic and compression functions, produced in a trusted foundry, can be integrated. For some applications, a 2D implementation is sufficient; however, other applications (e.g., Toshiba's Chip Scale Camera Module [25], see Section 6: Related Work) may require the high bandwidth and low latency only possible with a 3D implementation.

3. DESIGN GOALS

Our proposed architecture has two major goals: (1) high performance, comparable to that of other processors in the market; and (2) the ability for the control plane to gather and compress architectural state in the computation plane at runtime and send this compressed trace data off chip. It is impossible to track all registers in a processor due to the massive volume of data involved; therefore, one must carefully prioritize what data to monitor.

Mysore et al. [16] propose a 3D architecture for profiling that captures many different signals indicative of state and changes to state, e.g., memory addresses, memory values, program counter, opcodes, register names, cache misses, etc., in order to be sufficiently flexible for a variety of analysis techniques. This set of signals yields an estimate of the number of inter-die vias for sending the data to an analysis engine, and they estimate that to collect 1024 bits of profile data each cycle requires 1024 inter-die connections. Important signals to monitor include the control unit, program counter, status register, instruction register, and data addresses. We apply the results of their study to help us estimate the number of die-to-die vias required for our proposed design.

The following section describes the design choices necessary to achieve our design goals.

4. DESIGN CHOICES

This section describes the key design parameters for a 3D data transformation processor and a justification for each.

4.1 Manufacturing Process

The range of choices includes face-to-face bonding and face-to-back bonding. Decision factors include the number of layers required, level and ease of testing required, and via density. Our choice is to use face-to-face bonding because it provides testing advantages, greater via density, and the smallest possible distance between layers [3]. Also, since our design does not require more than two layers, face-to-back bonding is unnecessary.

4.2 Compression Algorithm/Hardware

Many compression algorithms and hardware implementations are available. The decision factors include the compression ratio possible for a given set of trace files, the area cost of the hardware implementation, and the needed throughput. The optimal compression algorithm and hardware depend on the type of trace. Our choice, based on the compression study described in Section 4.6, is to use two-stage compression. The first stage is *filtering*, and the second is general-purpose (gzip).

4.3 Cryptographic Algorithm/Hardware

The range of choices of cryptographic algorithm and hardware implementation includes a wide variety of cryptographic primitives and hardware implementations. The decision factors include security, the ability to support a variety of applications requiring cryptographic, area cost, and throughput. Our specific choice is to include units for AES-128, SHA-1, and SHA-512. These primitives support a wide variety of applications, e.g., networking, and they provide both security and speed.

4.4 Interface between Planes

The range of choices for the interface between the two planes includes whether to use a direct connection or a bus as well as the width of the connection (the number of wires corresponding to the number of vertical connections required). The decision factors include how well the selected technology supports speed, simplicity, cost, and density of vias. Our choice is to use 128 vias as a direct connection to send dynamic architectural state from the computation plane to the control plane, in order to access (using taps) the program counter (64 bits) and memory address registers (64 bits), which provide useful data for the dynamic analysis of the memory behavior of programs. To keep the number of vias manageable, this design does not support the tapping of all architectural state. We leave to future work the development of a general-purpose interface capable of supporting a wider range of program traces.

We also make the choice to use a 32-via direct connection to send the encrypted and compressed stream back down to the I/O interface in the computation plane (compression reduces the number of required vias to 32). A direct connection is faster and has lower cost than a bus, and our single producer, single consumer scheme does not need the contention-resolution provided by a bus.

4.5 Other Issues

We summarize the other main design considerations: (1) For the mechanism of coordination between the two planes and for the configuration/initialization of the control plane, we choose 8-bit (1-byte) control words, stored in special registers, along with a single via to signal the control plane when to act upon the control word. We base this choice on its simplicity and the small number of face-to-face vias required. (2) For the interface within the control plane between the compression and cryptographic coprocessors, the output of the compression circuit is connected to the input of the cryptographic circuit because compressing encrypted data does not yield good compression ratios [5]. (3) For the delivery of I/O and power to/from the outside world, we choose to employ the existing I/O capability of the computation plane rather than building a dedicated I/O controller in the control plane. We base our decision on its simplicity, low cost, and feasible number of vertical connections; however, we note that independent I/O and power delivery to/from the control plane would be useful from a security perspective, and we suggest this as a topic for future work. (4) For the computation plane hardware, we select a high-performance[1] general-purpose processor available in the marketplace in order to study real application workloads and realize the economic advantages of dual use of the computation plane; this requires modifying the CPU to support the optional attachment of a control plane [23]. (5) For clock synchronization, we choose the *tree network* method shown to be effective in previous research [18] to provide clock signals to the CPU, compression coprocessor, and encryption coprocessor, using three clock buffers between the two planes.

4.6 Compression Study

The goal of our compression study was to determine the optimal compression strategy for a set of real execution traces. We used TCgen [2], designed by Martin Burtscher specifically for generating lossless trace compressors from user-generated descriptions, to compress a set of trace files we generated using Pin [14]. Our trace files capture the memory access behavior of the Linux applications Firefox, Gimp, Mozilla, OpenOffice, and Opera, and they have fields for instruction count, program counter, memory address, and size. TCgen compresses each field individually rather than compressing all of the fields together.

For each field, we varied the parameters of TCgen, including the algorithm and the size of the data structures internal to each algorithm. Algorithms available in TCgen include Last Value (LV), Stride Predictor (ST), Finite-Context-Method (FCM), and Differential-Finite-Context-Method (DFCM). LV and ST use one internal table, but FCM and DFCM use two internal tables. Therefore, for FCM and DFCM, we vary the sizes (number of columns) of both tables, and for LV and ST we only vary one table.

	Number of Columns in Table (n)						
Algo.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
dfcm7[n]	7.22	5.42	5.41	5.40	5.98	6.02	5.99
dfcm6[n]	.578	.366	0	0	.004	.002	.002
dfcm5[n]	.956	.522	.522	.522	.526	.526	.526
dfcm4[n]	9.10	.002	0	0	0	0	0
dfcm3[n]	.016	2.08	2.08	2.08	2.12	2.12	2.12
dfcm2[n]	.062	.118	.232	.192	.698	.694	.692
dfcm1[n]	53.6	57.1	57.3	53.7	69.8	69.8	69.8
fcm7[n]	0	0	.188	0	0	0	0
fcm6[n]	0	0	0	0	0	0	0
fcm5[n]	0	0	0	.002	0	0	0
fcm4[n]	0	0	0	0	0	0	0
fcm3[n]	.004	0	0	0	0	0	0
fcm2[n]	.008	.006	.004	.012	.004	.004	.004
fcm1[n]	24.9	32.5	32.6	18.6	19.6	19.6	19.6
st[n]	0	0	0	0	0	0	0
lv[n]	0	0	0	0	.06	.06	.06
unpred.	3.62	1.88	1.72	19.5	1.25	1.2	1.17

Table 1: Number of correct predictions (%) for each configuration of TCgen while compressing the program counter field (average of all five trace files)

Since TCgen is prediction-based compression, the number of correct predictions indicates the effectiveness of the compression. Table 1 shows the number of correct predictions for each configuration of TCgen when compressing just the program counter field (average of all five trace files). The rows of this table correspond to different compression algorithms, and the columns correspond to the size of the algorithm's internal table (n). Algorithms include last value (LV), stride predictor (ST), finite-context-method (FCM), and differential-finite-context-method (DFCM).

LV and ST use one table; for algorithms that use two internal tables (FCM and DFCM), we vary the sizes of both tables. For example, fcm1[n] indicates the FCM algorithm, where its first table has one column and its second table has n columns. *Unpredictable* corresponds to those symbols in the trace that were never predicted correctly. For compressing the program counter field, the configuration with the greatest percentage (69.8%) uses DFCM with one column in its first internal table and five columns in its second internal table. We found that DFCM is also effective for the other fields.

After applying TCgen, we then apply a general-purpose compression algorithm (gzip) to compress the trace file further. On average, TCgen (the first stage) improves the compression ratio of gzip (the second stage) from 46.5 to 58.9 for this set of trace files. The design implication of our study is that the compression unit should use a two-stage compression, with the first using TCgen/DFCM and the second using gzip. Figure 1 shows the data in Table 1 graphically. Figure 2 shows the results when compressing the data address field. For this field, the differential property of DFCM does not contribute to the compression since data addresses do not have a fixed stride. Therefore, we recommend using the FCM algorithm for data addresses.

[1] Dissimilarity between the computation plane and the control plane presents significant engineering challenges, e.g., if the control plane uses a different technology node than the computation plane. It may be necessary to instantiate multiple instances of the compression hardware to keep up with the extremely fast computation plane and to carefully design the control plane buffers that receive data from the computation plane.

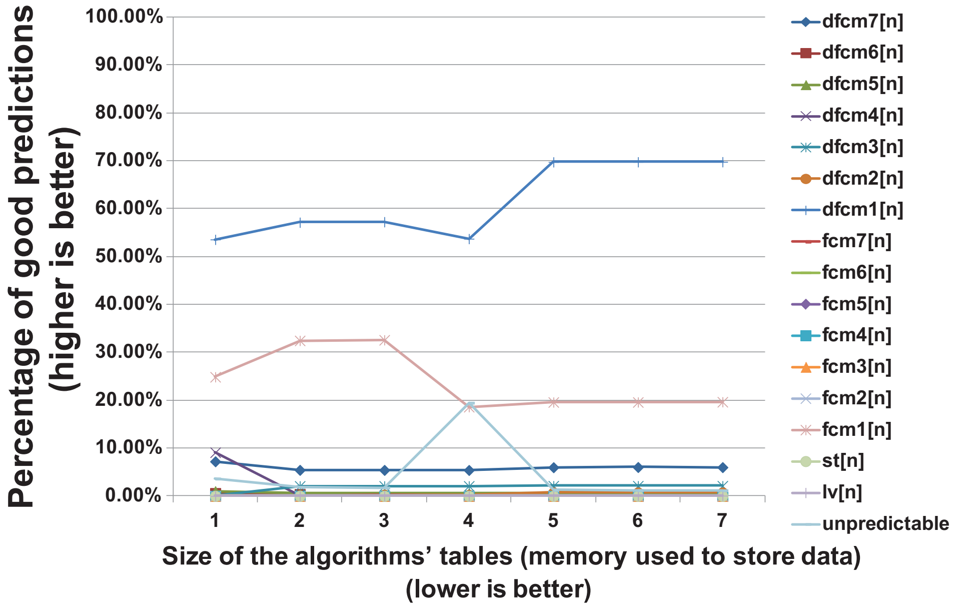

Figure 1: Number of correct predictions (%) for each configuration of TCgen while compressing the program counter field (average of all five trace files)

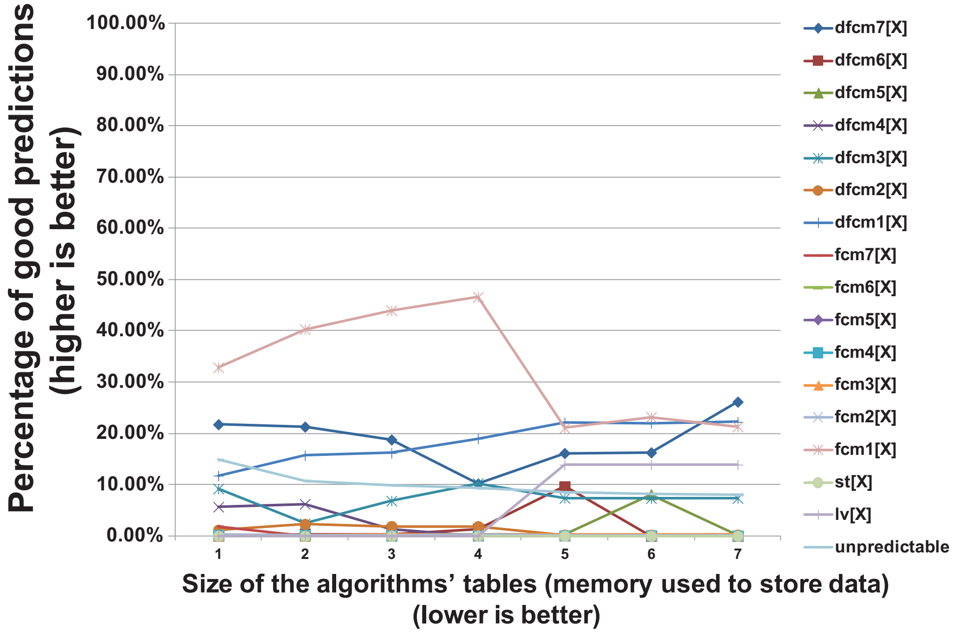

Figure 2: Number of correct predictions (%) for each configuration of TCgen while compressing the data address field (average of all five trace files)

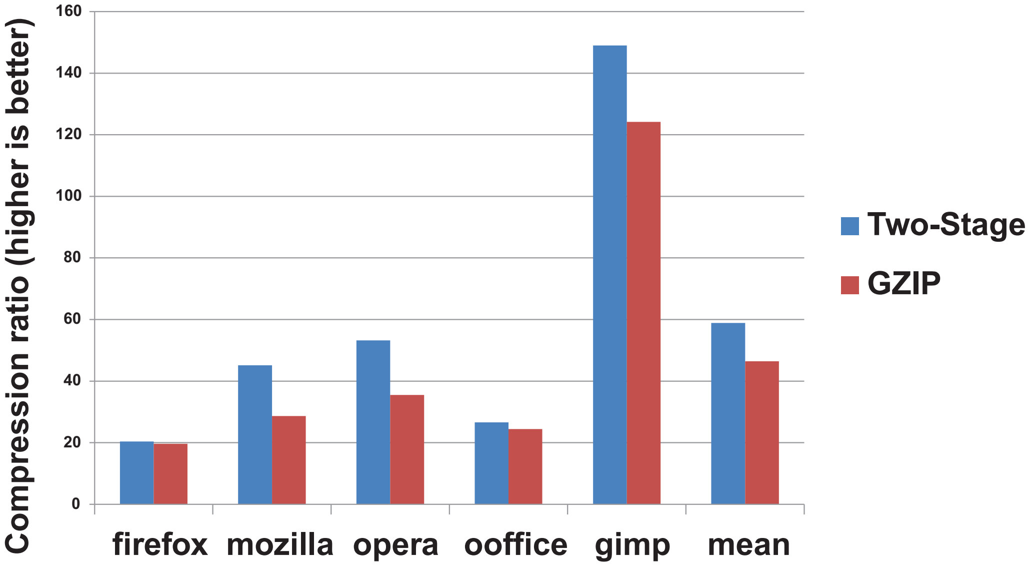

Figure 3: Compression ratio for the program counter field. Our two-stage proposal (DFCM + GZIP) has a slight advantage over a single GZIP stage.

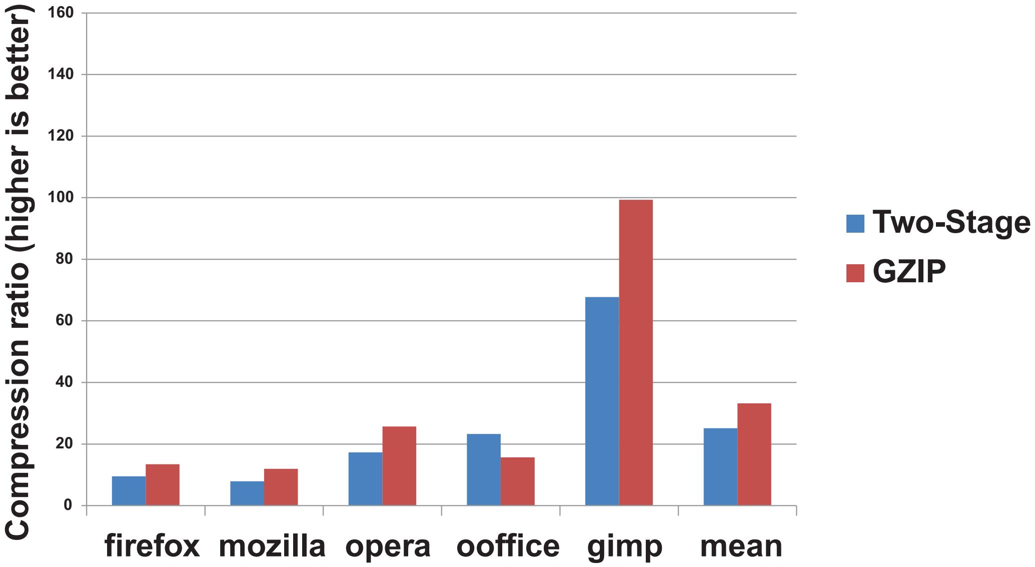

Figure 4: The low percentage of good prections in the data address field results in a poor compression ratio for our proposed design. The first stage's algorithm must be carefuly chosen in order to achieve a better compression ratio.

5. SYSTEM ARCHITECTURE

In this section, we apply the results of our design choices and our compression study to build a 3D transformation engine. We use the following circuit-level primitives [23] to allow the control plane to interact with the computation plane: disabling, tapping, rerouting, and overriding. This architecture requires 128 direct links (128 bits) between the computation and control planes to access (using taps) the program counter (64 bits) and memory address registers (64 bits). These direct links lead from various locations in the computation plane to the compression circuit in the control plane.

Figure 5 depicts a block diagram of the computation plane, showing the control unit of the microprocessor, program counter, memory address register, cache, clock (for synchronizing the two planes), I/O interface, and I/O controller (to send the compressed stream off-chip). The compressed and encrypted stream is sent back to the computation plane along another set of face-to-face vias to eliminate the need for a separate I/O capability of the control plane. As the computation plane may be considered an unsafe environment, we provide the option of encrypting the trace file before it is exported.

The main components of the control plane are the microprocessor interface (Figure 10), the compression coprocessor (Figure 7), and the cryptographic coprocessor (Figure 6). The control plane also uses buffers to ensure that compression proceeds smoothly without stalling the processor or dropping data [15]. In addition to transferring data between layers, clock signals and query/control signals must also be transmitted. The microprocessor interface (Figure 10) in the control plane manages the query and control signals, which include: a clock signal, a read/write signal, an address/data byte, and externally accessible registers to send/receive the signals. The registers include error, status, interrupt, command, and reset. Figure 9 shows the integration of the computation plane, microprocessor interface, compression unit, and cryptographic unit into a full system.

The read/write signal uses one or zero to indicate a read or write. The address/data signals use one byte. The two most significant bits address the interface register, the next bit specifies whether the signal is for the compression or cryptographic hardware, and the last five bits are the instruction, supporting 32 query/control instructions for each coprocessor. For synchronization of the CPU in the computation plane and the two coprocessors in the control plane, we use a three-level buffer clock distribution network, which helps reduce transmission time [6, 18]. The compression coprocessor uses Content Addressable Memories (CAM), which allow multiple comparisons to be made in parallel [17].

The compression processor uses two-stage compression: the first uses the Differential Finite Context Method (DFCM) of TCgen [2]; the second stage uses gzip. Figure 8 shows a block diagram of the gzip module. A FIFO buffer is used to avoid stalling the processor and to smooth out speed variations due to warm up time. The 64-bit output is sliced into 32 bits prior to being sent to the encryption unit, i.e., each 64-bit value is split into two 32-bit values.

The encryption unit is inspired by the HSSec cryptographic coprocessor [10]. It supports AES-128, SHA-1, and SHA-512, selected for their security, speed, low power, and their ability to support a variety of applications. The control unit manages data processing and communication with the compression processor and microprocessor interface. The AES-128, SHA-1, and SHA-512 units use a common 64-bit global data bus. The key scheduler is used for key expansion and generating message schedules. The memory block consists of a register file, padding unit, and S-boxes. The mode interface is responsible for modifying the input to the cryptographic primitives. The key scheduler performs the RotWord and SubWord transformations and provides constants needed by the hash functions: SHA-1 uses a sequence of 80 32-bit words, and SHA-1 uses a sequence of 80 64-bit words.

6. RELATED WORK

Vasudevan et al. have developed the XTRec primitive for recording the instruction-level execution trace of a commodity computing system while simultaneously ensuring the integrity of the recorded information on commodity platforms without requiring software modifications or specialized hardware [24]. Such a primitive can be used to perform postmortem analysis for forensic purposes. Our work differs from XTRec in that we are proposing a specialized 3DIC approach, and we argue that our proposed sytem would facilitate the capture of additional activity besides the instruction trace, at higher bandwidth, in exchange for the higher cost of specialized hardware.

Many 3D applications have been built successfully, including 3D chips for imaging [25], medicine [9], particle physics [4], reconfigurable hardware [20], and high-performance microprocessors [1], [19], [13], [12], [11]. Previous work on security applications of 3D integration includes a 3D design for mitigating access-driven cache side channel attacks (and the circuit-level primitives needed to support such designs) [23]; a study of whether individual layers must be independently trustworthy for the system of joined dies to provide certain trustworthy functions [7]; additional primitives and design flow modifications to support security in 3D designs [8], and a qualitative security analysis of a new class of 3-D crypto coprocessors [22]. This paper builds on [22] by presenting a specific instance of a data transformation processor that combines cryptography and compression.

7. CONCLUSION

We have presented an architecture for a 3D data transformation processor and a rationale for each of the key design decisions, including a compression study that determined the optimal compression algorithm for a specific set of traces generated using the Pin dynamic binary instrumentation tool. We leave to future work the hardware implementation, simulation, FPGA prototype, and 3D IC realization of the design in silicon.

8. ACKNOWLEDGMENTS

This research was funded by National Science Foundation Grant CNS-0910734. The views expressed in this paper do not represent the Department of Defense or National Science Foundation.

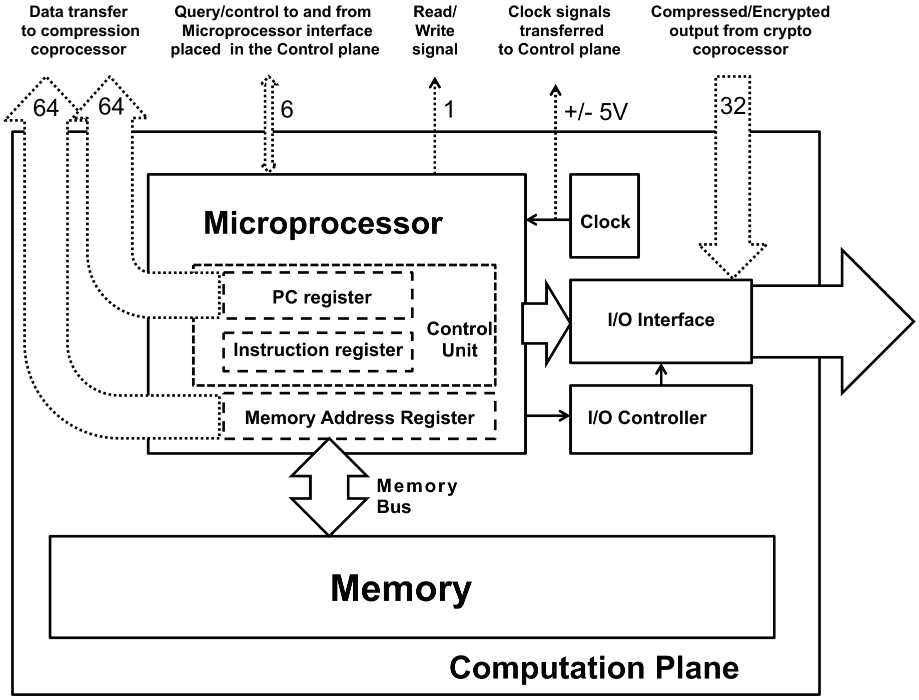
Data transfer to compression coprocessor
Query/control to and from Microprocessor interface placed in the Control plane
Read/ Write signal
Clock signals transferred to Control plane
Compressed/Encrypted output from crypto coprocessor
64
64
6
1
+/- 5V
32
Microprocessor
Clock
PC register
Instruction register
Control Unit
I/O Interface
Memory Address Register
I/O Controller
Memory Bus
Memory
Computation Plane

Figure 5: Block diagram of computation plane

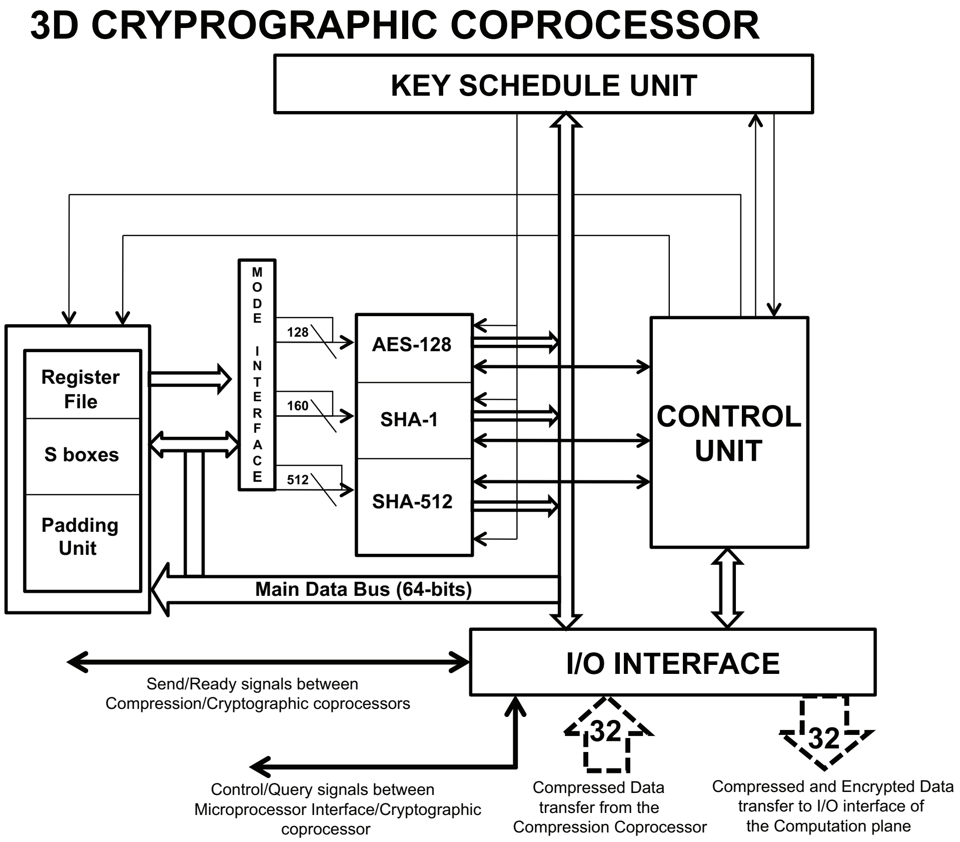
3D CRYPROGRAPHIC COPROCESSOR
KEY SCHEDULE UNIT
MODE INTERFACE
128
160
512
AES-128
SHA-1
SHA-512
Register File
S boxes
Padding Unit
CONTROL UNIT
Main Data Bus (64-bits)
I/O INTERFACE
Send/Ready signals between Compression/Cryptographic coprocessors
Control/Query signals between Microprocessor Interface/Cryptographic coprocessor
32
Compressed Data transfer from the Compression Coprocessor
32
Compressed and Encrypted Data transfer to I/O interface of the Computation plane

Figure 6: Block diagram of the cryptographic unit, after [10]

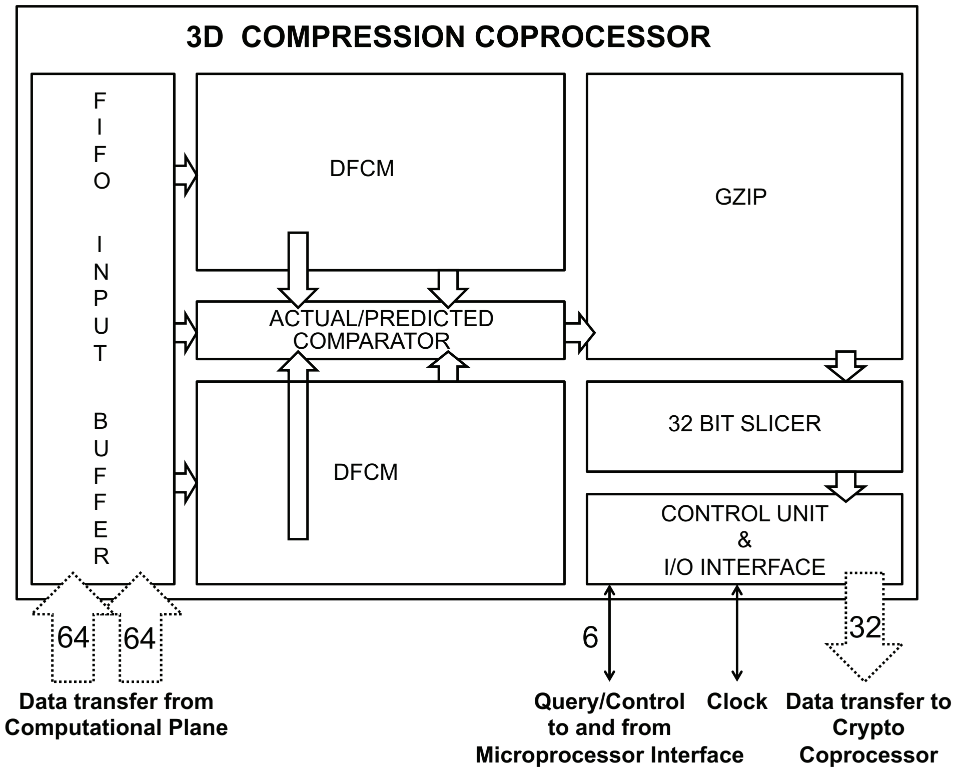
3D COMPRESSION COPROCESSOR
FIFO INPUT BUFFER
DFCM
ACTUAL/PREDICTED COMPARATOR
DFCM
GZIP
32 BIT SLICER
CONTROL UNIT & I/O INTERFACE
64
64
Data transfer from Computational Plane
6
Query/Control to and from Microprocessor Interface
Clock
32
Data transfer to Crypto Coprocessor

Figure 7: Block diagram of the compression unit

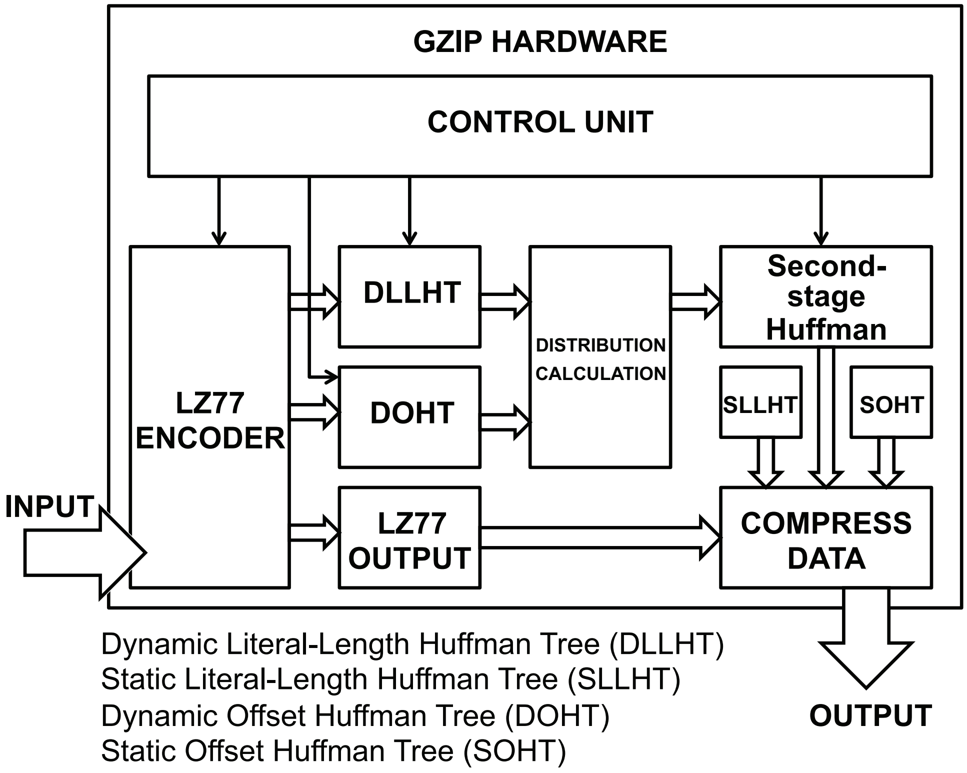
GZIP HARDWARE
CONTROL UNIT
LZ77 ENCODER
DLLHT
DOHT
LZ77 OUTPUT
DISTRIBUTION CALCULATION
Second-stage Huffman
SLLHT
SOHT
COMPRESS DATA
INPUT
OUTPUT
Dynamic Literal-Length Huffman Tree (DLLHT)
Static Literal-Length Huffman Tree (SLLHT)
Dynamic Offset Huffman Tree (DOHT)
Static Offset Huffman Tree (SOHT)

Figure 8: Block diagram of gzip unit, after [21]

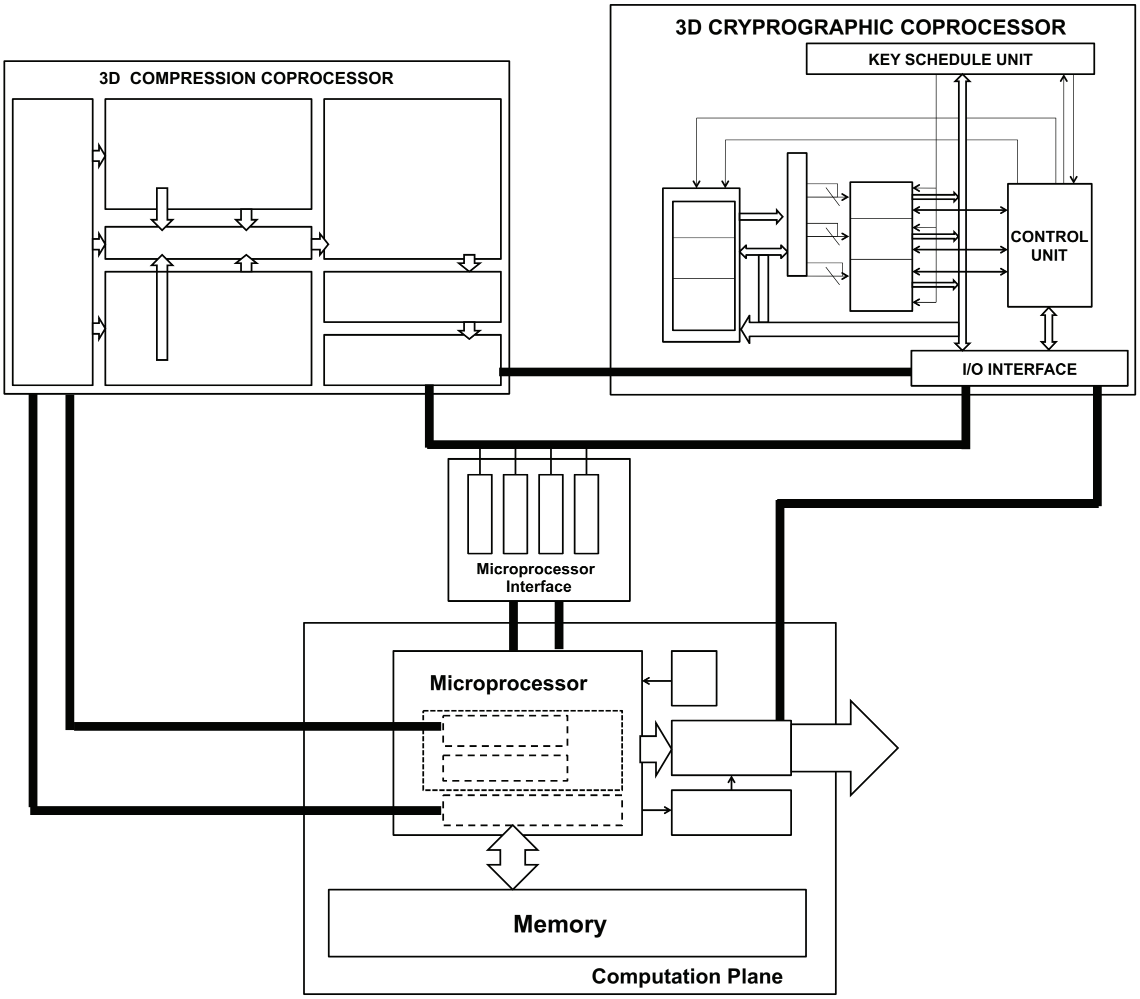

Figure 9: Block diagram of the full system

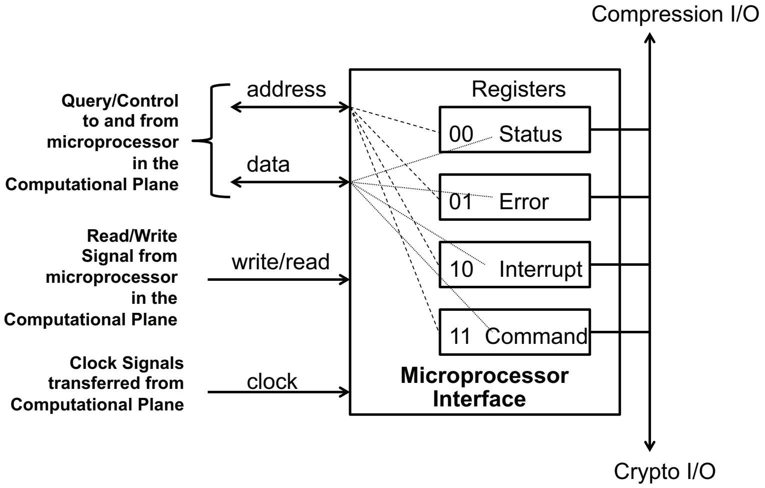
Compression I/O
Query/Control to and from microprocessor in the Computational Plane
address
data
Registers
00 Status
01 Error
10 Interrupt
11 Command
Read/Write Signal from microprocessor in the Computational Plane
write/read
Clock Signals transferred from Computational Plane
clock
Microprocessor Interface
Crypto I/O

Figure 10: Block diagram of the microprocessor interface unit, which manages the query and control signals. This interface receives a clock signal, read/write signal, and address/data byte; it also has externally accessible registers to send and receive the signals.

9. REFERENCES

[1] B. Black, M. Annavaram, N. Brekelbaum, J. DeVale, L. Jiang, G. H. Loh, D. McCaule, P. Morrow, D. W. Nelson, D. Pantuso, P. Reed, J. Rupley, S. Shankar, J. Shen, and C. Webb. Die stacking (3D) microarchitecture. In *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Orlando, FL, December 2006.

[2] M. Burtscher. TCgen 2.0: a tool to automatically generate lossless trace compressors. *ACM SIGARCH Computer Architecture News*, 34(3), June 2006.

[3] W. R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A. M. Sule, M. Steer, and P. D. Franzon. Demystifying 3D ICs: The pros and cons of going vertical. *IEEE Design and Test of Computers*, 22(6), November/December 2005.

[4] M. Demarteau, Y. Arai, H.-G. Moser, and V. Re. Developments of novel vertically integrated pixel sensors in the high energy physics community. In *IEEE International Conference on 3D System Integration (3DIC)*, San Francisco, CA, September 2009.

[5] R. Elbaz, L. Torres, G. Sassatelli, P. Guillemin, C. Anguille, M. Bardouillet, C. Buatois, and J. Rigaud. Hardware engines for bus encryption: A survey of existing techniques. In *Proceedings of the Conference on Design, Automation, and Test in Europe (DATE)*, Munich, Germany, March 2005.

[6] E. G. Friedman. Clock distribution networks in synchronous digital integrated circuits. *Proceedings of the IEEE*, 89(5), May 2001.

[7] T. Huffmire, T. Levin, M. Bilzor, C. E. Irvine, J. Valamehr, M. Tiwari, T. Sherwood, and R. Kastner. Hardware trust implications of 3-D integration. In *Proceedings of the 5th Workshop on Embedded Systems Security (WESS)*, Scottsdale, AZ, October 2010.

[8] T. Huffmire, T. Levin, C. Irvine, R. Kastner, and T. Sherwood. 3-D extensions for trustworthy systems. In *Proceedings of the International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA)*, Las Vegas, NV, USA, July 2011.

[9] Y. Kaiho, Y. Ohara, H. Takeshita, K. Kiyoyama, K.-W. Lee, T. Tanaka, and M. Koyanagi. 3D integration technology for 3D stacked retinal chip. In *IEEE International Conference on 3D System Integration*, San Francisco, CA, September 2009.

[10] A. Kakarountas, H. Michali, C. Goutis, and C. Efstathiou. Implementation of HSSec: a high-speed cryptographic co-processor. In *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Patras, Greece, September 2007.

[11] J. Kim, C. Nicopoulos, D. Park, R. Das, Y. Xie, N. Vijaykrishnan, M. S. Yousif, and C. R. Das. A novel dimensionally-decomposed router for on-chip communication in 3D architectures. In *Proceedings of the 34th International Symposium on Computer Architecture*, San Diego, CA, June 2007.

[12] G. H. Loh. 3-D stacked memory architectures for multi-core processors. In *International Symposium on Computer Architecture (ISCA)*, Beijing, China, June 2008.

[13] G. H. Loh, Y. Xie, and B. Black. Processor design in 3D die-stacking technologies. *IEEE Micro*, 27(3), May-June 2007.

[14] C.-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V. J. Reddi, and K. Hazelwood. Pin: Building customized program analysis tools with dynamic instrumentation. In *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, Chicago, IL, USA, June 2005.

[15] M. Milenkovic, A. Milenkovic, and M. Burtscher. Algorithms and hardware structures for unobtrusive real-time compression of instruction and data address traces. In *Proceedings of the Data Compression Conference (DCC)*, Snowbird, UT, USA, March 2007.

[16] S. Mysore, B. Agrawal, S. Lin, N. Srivastava, K. Banerjee, and T. Sherwood. Introspective 3-D chips. In *Proceedings of the 12th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, San Jose, CA, October 2006.

[17] J. L. Nunez and S. Jones. Gbit/s lossless data compression hardware. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 11(3), June 2003.

[18] V. F. Pavlidis, I. Savidis, and E. G. Friedman. Clock distribution networks for 3-D integrated circuits. In *Proceedings of the IEEE Custom Integrated Circuits Conference (CICC)*, San Jose, CA, September 2008.

[19] K. Puttaswamy and G. Loh. Thermal analysis of a 3D die-stacked high-performance microprocessor. In *Proceedings of the 16th ACM Great Lakes Symposium on VLSI (GLSVLSI'06)*, Philadelphia, PA, May 2006.

[20] S. A. Razavi, M. S. Zamani, and K. Bazargan. A tileable switch module architecture for homogeneous 3D FPGAs. In *Proceedings of the IEEE International Conference on 3D System Integration*, San Francisco, CA, September 2009.

[21] S. Rigler. FPGA-based lossless data compression using GNU zip, 2007.

[22] J. Valamehr, T. Huffmire, C. Irvine, R. Kastner, C. K. Koc, T. Levin, and T. Sherwood. A qualitative security analysis of a new class of 3-D integrated crypto co-processors. *Festschrift Jean-Jacquest Quisquarter, Lecture Notes in Computer Science*, 6805, 2011.

[23] J. Valamehr, M. Tiwari, T. Sherwood, A. Arfaee, R. Kastner, T. Huffmire, C. Irvine, and T. Levin. Hardware assistance for trustworthy systems through 3-D integration. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, Austin, TX, December 2010.

[24] A. Vasudevan, N. Qu, and A. Perrig. XTRec: Secure real-time execution trace recording on commodity platforms. In *Hawaii International Conference on System Sciences (HICSS)*, Kauai, HI, January 2011.

[25] H. Yoshikawa, A. Kawasaki, T. Iiduka, Y. Nishimura, K. Tanida, K. Akiyama, M. Sekiguchi, M. Matsuo, S. Fukuchi, and K. Takahashi. Chip scale camera module (CSCM) using through-silicon via (TSV). In *IEEE International Solid-State Circuits Conference (ISSCC)*, San Francisco, CA, February 2009.